

HTML, CSS & JS basis

Les 6:

HTML formulieren CSS responsive



Onderwerpen les:

- HTML form elements
- Form input en JS
- Form validatie
- CSS responsive media queries
- CSS responsive met flexbox
- CSS responsive met Grid

Leerdoelen

- ✓ Aan het einde van de dag ken je verschillende type input velden
- ✓ Aan het einde van de dag kan je input values ophalen en gebruiken in JavaScript
- ✓ Aan het einde van de dag kan je form validation toepassen
- ✓ Aan het einde van de dag kan je een website responsive maken
- ✓ Aan het einde van de dag weet je wat Grid is

HTML form elemens

<form>

De **<form>** tag wordt gebruikt om aan te geven dat het element en de child elements een formulier betreft. Hierin kun je alles gebruiken om je formulier vorm te geven zoals headings, divs, paragraven etc. Ook kun je hierin form elementen gebruiken zoals: inputs, radio buttons, checkboxes, selects en een submit button.

Een form tag kan veel attributen hebben. Wij gaan hiervan action en onsubmit behan- den. Andere veel gebruikte attributen zijn: method en target

HTML form elemens

<form>

action

URL waar de data heen wordt gestuurd. Vul je dit niet in? dan wordt dit de huidige pagina en herlaadt de pagina bij het versturen. Voorbeeld hieronder schakelt dit uit.

onsubmit

De functie die hier is geplaatst vuurt af zodra er op de submit button wordt geklikt.

```
<form action="javascript:void(0);" onsubmit="formFunction()">
  <input class="button" type="submit" value="Versturen">
</form>
```

HTML form elemens

form elementen

In een formulier wil je meestal input ontvangen van een bezoeker. Dit kan zijn:

- een stuk tekst zoals een naam, wachtwoord, email adres.
- een keuze uit een selectie zoals favoriete snack: Mars, Snickers of Bounty
- Een keuze waarbij meerdere keuzes mogelijk zijn zoals welke kleuren vind je mooi? Rood, blauw, groen, paars, geel.
- Een keuze waarbij 1 mogelijk is. Ga je akkoord met... ja of nee
- en nog veel meer.

Voor al deze types van input zijn er verschillende elementen die je kan gebruiken.

| https://www.w3schools.com/html/html_form_elements.asp

HTML form elemens

<input>

HTML kent veel type input velden zoals: text, number, date, color. Zie w3schools voor de volledige lijst. Een input heeft vaak de attributen type, id en name.

```
<input type="text" id="voornaam" name="voornaam">
<input type="number" id="leeftijd" name="leeftijd">
```

| https://www.w3schools.com/html/html_form_input_types.asp

HTML form elemens

«label»

De label tag wordt gebruikt om aan te geven waar een form element(en) voor zijn. De label tag heeft een for attribute. Deze geeft aan voor welk element het label is. Deze zal gelijk zijn aan de id van dit element

```
<label for="voornaam">Voornaam</label>
<input type="text" id="voornaam" name="voornaam">
```

| https://www.w3schools.com/tags/tag_label.asp

«fieldset en legend»

Een fieldset wordt gebruikt om gerelateerde form elementen te groeperen. Een legend is een label voor een gegroepeerde elementen.

| https://www.w3schools.com/tags/tag_fieldset.asp

HTML form elemens

<input type="checkbox">

Een bezoeker kan een checkbox niet of wel aanvinken. Heb je meerdere checkboxes in een groep? dan kan de gebruiker 0 of meerdere checkboxes aanvinken.

```
<legend>Coding languages you have learned:</label>
<input type="checkbox" id="html" name="coding_language" value="html" checked>
<label for="html">HTML</label>
<input type="checkbox" id="java" name="coding_language" value="java">
<label for="java">JAVA</label>
```

- | https://www.w3schools.com/tags/att_input_type_checkbox.asp
- | <https://html5-tutorial.net/forms/checkboxes/>

HTML form elemens

<input type="radio">

Een bezoeker kan een radio button niet of wel aanvinken. Heb je meerdere buttons in een groep? dan kan de gebruiker 0 of 1 radio button aanvinken.

```
<legend>What is your favorite candy?</label>
<input type="radio" id="mars" name="fav_candy" value="mars">
<label for="mars">Mars</label>
<input type="radio" id="bounty" name="fav_candy" value="bounty">
<label for="bounty">Bounty</label>
```

- | https://www.w3schools.com/tags/att_input_type_radio.asp
- | <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/radio>

HTML form elemens

«**select**»

Een lijst met keuzes in een dropdown.

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

| https://www.w3schools.com/tags/tag_select.asp

HTML input en JS

get value

value van input (text, number en meer) en select

We gaan de waarde ophalen van een input veld dat niet behoort tot een groep zoals meerdere checkbox velden of radio buttons.

Aangezien we al onze input velden netjes een id geven kunnen we het element heel makkelijk ophalen met `document.getElementById('nameid')`. Echter willen we niet het element ophalen maar slechts de value van het element. De value van een text type input veld zal de door de gebruiker ingetypde waarde zijn.

```
let voornaam = document.getElementById('voornaam').value;
```

HTML input en JS

get value

value checkbox

De waarde ophalen van een aangevinkte checkbox kan op meerdere manieren. Wij gaan gebruikmaken van een querySelector. Hierin kan je zoeken naar een id, class, tag, maar ook naar een tagnaam met een bepaalde naam voor het attribuut name of attribuut id. Vervolgens kunnen we angeven dat het element dat wij zoeken ook 'checked moet zijn.

```
let keuze = document.querySelector('input[name="verhaal"]:checked').value;
```

HTML input en JS

get value

ophalen aantal aangevinkte checkboxes

Misschien wil je van een groep weten welke aangevinkt zijn of misschien wel hoeveel checkboxes.

Stap 1: vind alle checkboxes binnen de groep en plaats deze in een variabele.

Stap 2: Maak een counter.

Stap 3: Maak een for loop die itereert over de checkboxes groep.

Stap 4: In de for loop een if statement -> indien de checkbox aangevinkt is counter + 1.

```
let keuze = document.querySelector('input[name="verhaal"]:checked').value;
```

HTML input en JS

get value

ophalen aantal aangevinkte checkboxes

```
let checkboxes = document.querySelectorAll('input[name="coding_languages"]');
let counter = 0;
for (let i = 0; i < checkboxes.length; i++) {
  if (checkboxes[i].checked) {
    counter++;
  }
}
```

HTML input en JS

get value

ophalen value radio buttons

Bij het ophalen van de waarde van de aangevinkte radio button binnen een groep doen we met de querySelector aan de hand van de tag input de name (naam groep) en hij moet checked zijn.

```
let keuze = document.querySelector('input[name="gender"]:checked').value;
```

Form validation

Client-side validation

Ingebouwde form validation

Met HTML form elementen en attributen is het mogelijk om form validatie toe te passen door aan te geven dat een veld verplicht is, een email moet bevatten, een min of max waarde moet hebben en meer.

JavaScript

Doormiddel van JavaScript code kun je eigen validatie schrijven. Ook hier kun je angeven dat een veld verplicht is, maar ook de gebruiker verplichten dat input moet voldoen aan een bepaald patroon of dat input moet overeenkomen met andere input (herhalen wachtwoord) en nog veel meer.

| https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation

Form validation

Ingebouwde form validation

required	Geeft aan dat een form field verplicht is om in te vullen voordat het formulier verstuurd kan worden.
minlength maxlength	Geeft de minimum en maximum lengte van text data aan.
min max	Geeft de minimum en maximum waarde aan van numerieke data.
type	Geeft aan wat voor waarde ingevoerd moet worden (email, number, url, date en meer).
pattern	Een patroon (regular expression) waaraan de input moet voldoen. Bijvoorbeeld alleen letters en nummers met niet meer dan 15 karakters: [A-Za-z0-9_]{1,15}

Form validation

Ingebouwde form validation

```
<input type="text" id="voornaam" name="voornaam" minlength=5 required>

<input type="password" id="password" name="password"
pattern="[A-Za-z0-9_]{1,15}" required>
```

Form validation

JavaScript form validation

Met JavaScript kun je voorkomen dat een formulier wordt gesubmit als het niet voldoet aan de voorwaarden die jij aan input velden mee geeft. Zodra iemand op de submit button klikt start een JavaScript functie om de input te controleren. Zodra de input niet voldoet 1 van deze voorwaarden zal de default actie van submit niet worden uitgevoerd.

JavaScript addEventListener

Met de addEventListener in JavaScript kun je code laten uitvoeren zodra een bepaald event gebeurt zoals een click, mouseover, submit etc. Dit event kun je makkelijk meegeven aan een arrow function (zie link beneden). Vervolgens zullen we in deze functie onze input voorwaarden controleren.

- | <https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener>
- | https://www.w3schools.com/js/js_arrow_function.asp

Form validation

JavaScript form validation

`preventDefault()`

De `preventDefault()` methode vertelt de browser dat het event de default actie niet moet uitvoeren. Voorbeeld: klikken op een `a` tag (link) met `event.preventDefault()` zal de browser de link niet openen. In ons geval willen wij het event submit niet uitvoeren zodra iemand op de submit button klikt maar niet voldoet aan onze input voorwaarden.

`if statements`

In een `if statement` bepalen we de voorwaarden voor onze input velden.

| <https://developer.mozilla.org/en-US/docs/Web/API/Event/preventDefault>

Form validation

JavaScript form validation

error messages

In onze functie plaatsen we een lege array om error messages op te slaan. Onderaan onze functie controleren we met een if statement of deze niet meer leeg is. Is deze niet leeg dan wordt de event.preventDefault() getriggerd en tonen we de foutmeldingen in een lege div.

- | https://www.w3schools.com/js/js_arrays.asp
- | <https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML>
- | <https://developer.mozilla.org/en-US/docs/Web/API/HTMLElement/innerText>

Form validation

JavaScript form validation

```
form.addEventListener('submit', (e) => {
  let messages = []
  if (fname.value === "" || fname.value == null) {
    messages.push('Name is required')
  }

  if (password.value.length <= 6) {
    messages.push('Password must be longer than 6 characters')
  }

  if (password.value.length >= 20) {
    messages.push('Password must be less than 20 characters')
  }

  if (password.value === 'password') {
    messages.push('Password cannot be password')
  }

  if (messages.length > 0) {
    e.preventDefault()
    errorElement.innerText = messages.join('\n ')
  }
})
```

CSS responsive

Responsive media queries

Wij gaan media queries gebruiken om er voor te zorgen dat de pagina's er goed uit zien op verschillende formaten schermen (computer, laptop, tablet en smart-phone).

In deze query geven we aan voor welk scherm formaat (width) onze code moet worden toegepast.

```
@media screen and (min-width: 480px) {  
    /* CSS hier */  
}
```

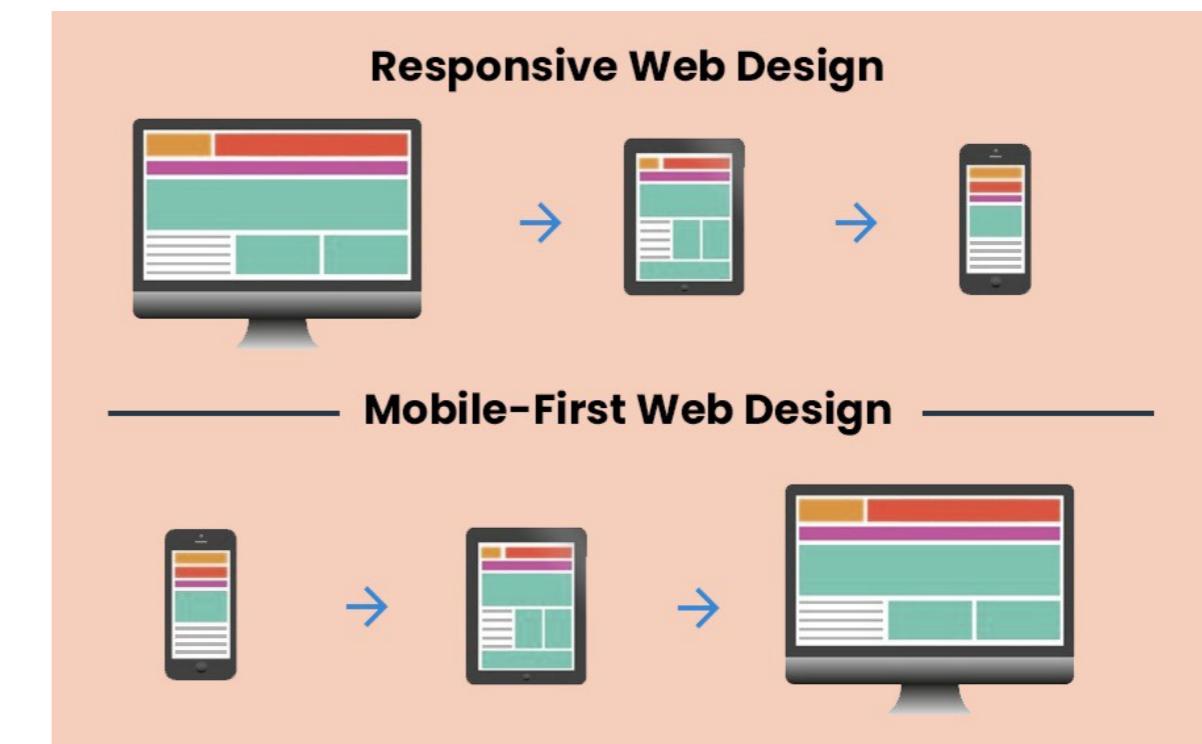
- | https://www.w3schools.com/css/css3_mediaqueries.asp
- | https://www.w3schools.com/cssref/css3_pr_mediaquery.php

CSS responsive

Responsive media queries

min of max width

De basis css boven de media queries kan de styling zijn voor grote schermen (laptops en desktops) of juist voor de kleine schermen (mobiel). Vanuit daar ga je de website responsive maken. Indien je van groot naar klein werkt zul je max width gebruiken om naar de kleinere schermen toe te werken. Indien je werkt vanuit de kleine schermen naar grote schermen zul je met min width werken. De laatste optie noemen we ook wel mobile first.



CSS responsive

Responsive media queries

```
.item {width: 25%;}

@media screen and (max-width: 768px) {
    .item {width: 50%;}
}

@media screen and (max-width: 425px) {
    .item {width: 100%;}
}
```

CSS responsive

Responsive met flexbox

flex-container wrap en no wrap

Met flex-wrap: wrap zullen items indien er niet genoeg ruimte in een regel is wrappen naar een volgende regel. Met flex-wrap: no wrap zullen items doorgaan op dezelfde regel ookal is er niet meer voldoende ruimte. Hierdoor krijg je een overflow.

flex-items grow en shrink

Met grow kun je ervoor zorgen dat items groeien zodat de hele regel wordt benut. Met flex shrink kun je ervoor zorgen dat items wel of niet krimpen afhankelijk van de grootte van de rij en de grootte van andere elementen in die rij.

CSS responsive

Responsive met flexbox

flex-items flex basis

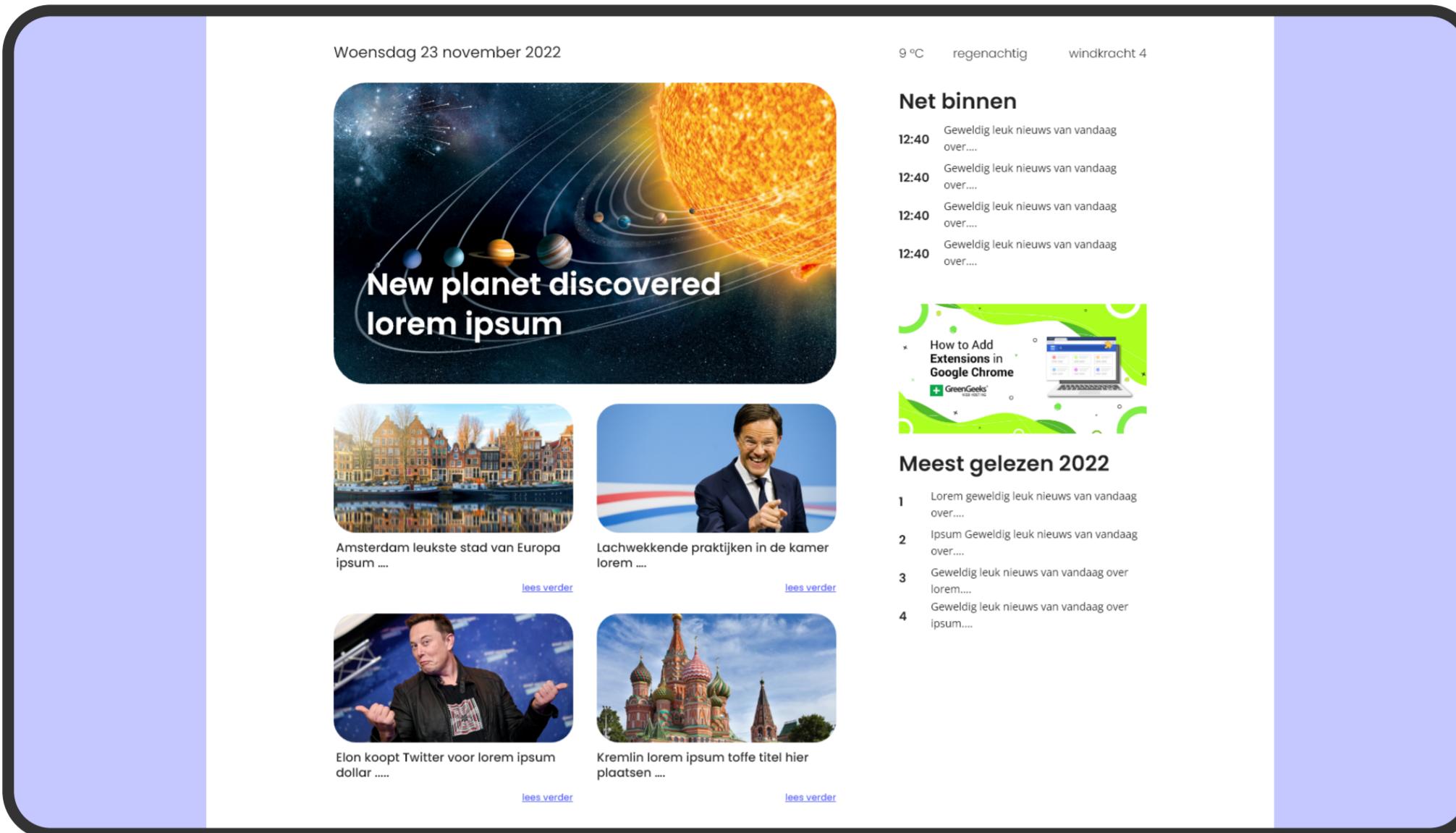
Met **flex-basis**: 150px of een andere width kun je aangeven wat de standaard breedteis van het element voordat de **flex-grow** en **flex-shrink** hun werk doen. Echter doet de width property precies hetzelfde indien de direction row is.

Voorbeeld: Je hebt een lijst met vakantiehuisjes. Op grote schermen heb je 4 naast elkaar op tablets 2 en op mobiel 1 huis per rij. Hier kan je de elementen een width van 25, 50 en 100% geven.

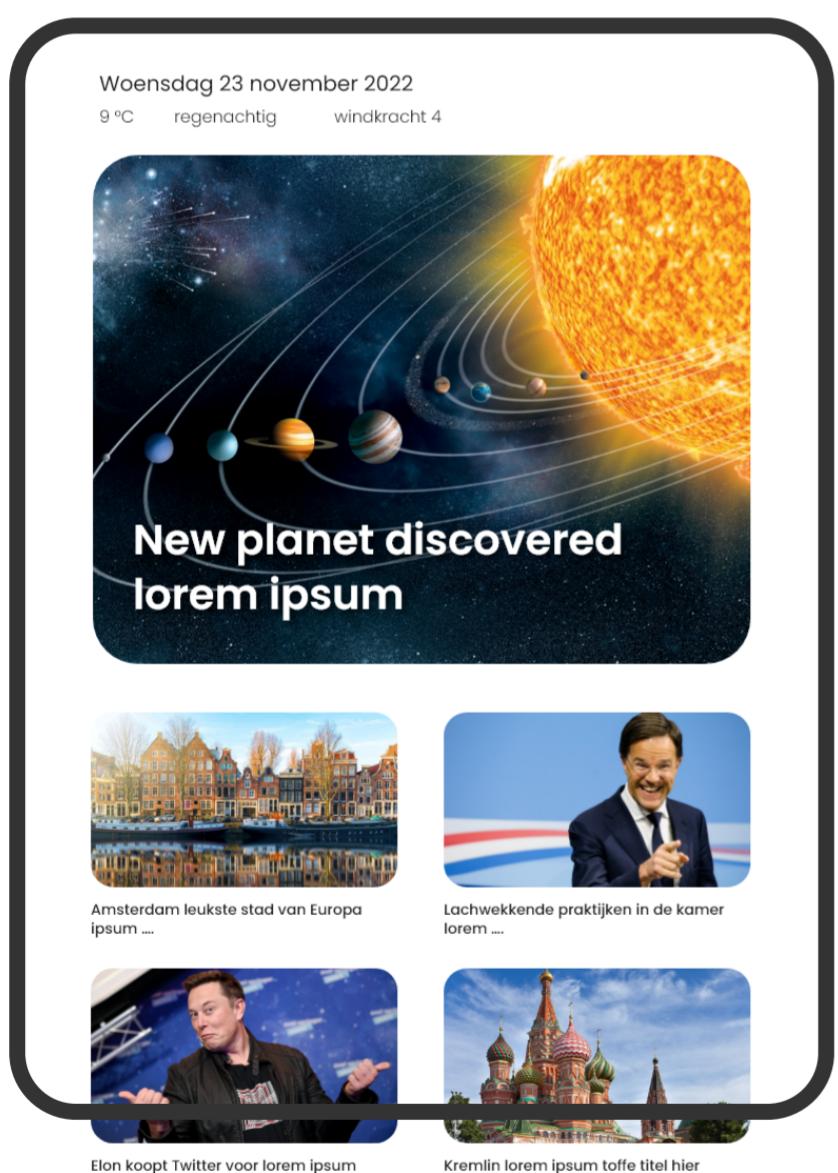
| https://www.w3schools.com/css/css3_flexbox_responsive.asp

CSS responsive

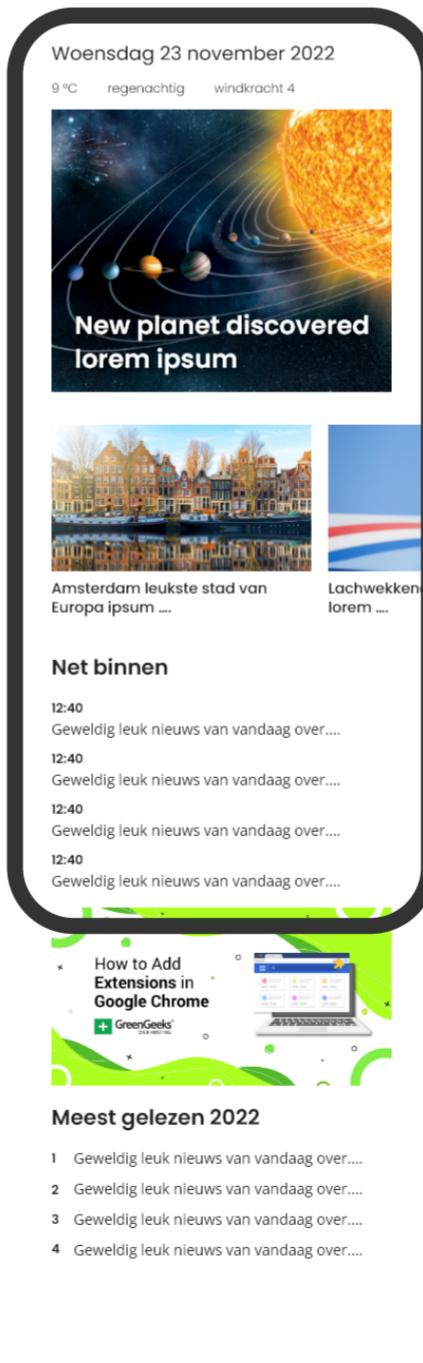
Responsive met flexbox



@media screen and (max-width: 768px) {}



@media screen and (max-width: 425px) {}



Net binnin
12:40 Geweldig leuk nieuws van vandaag over....
12:40 Geweldig leuk nieuws van vandaag over....
12:40 Geweldig leuk nieuws van vandaag over....
12:40 Geweldig leuk nieuws van vandaag over....

Meest gelezen 2022
1 Geweldig leuk nieuws van vandaag over....
2 Geweldig leuk nieuws van vandaag over....
3 Geweldig leuk nieuws van vandaag over....
4 Geweldig leuk nieuws van vandaag over....

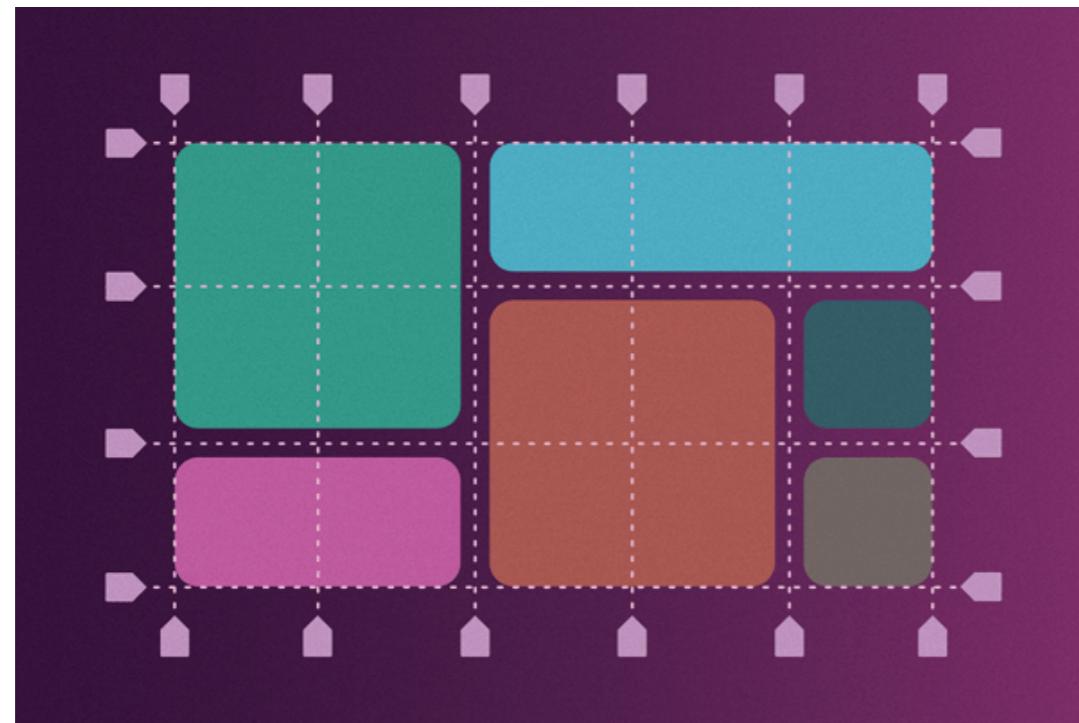


CSS responsive

Responsive met grid

Wat is Grid?

Grid is een display instelling net als inline, block, flex etc. Het is een tweedimensionaal raster-gebaseerd layout systeem. Met Grid kun je complexe layouts maken.



| <https://css-tricks.com/snippets/css/complete-guide-grid/>

CSS responsive

Responsive met grid

Grid termen

Grid Container

Het element met display: grid. Net als bij flexbox heb je een container en de items. De Grid Container is de parent van de Grid Items.

Grid Item

De children van de Grid Container. Alleen direct childs van de Grid Container worden Grid Items.

```
.grid {  
    display: grid;  
}
```

```
<div class="grid">  
    <div class="grid-item"></div>  
    <div class="grid-item"></div>  
    <div class="grid-item"></div>  
</div>
```

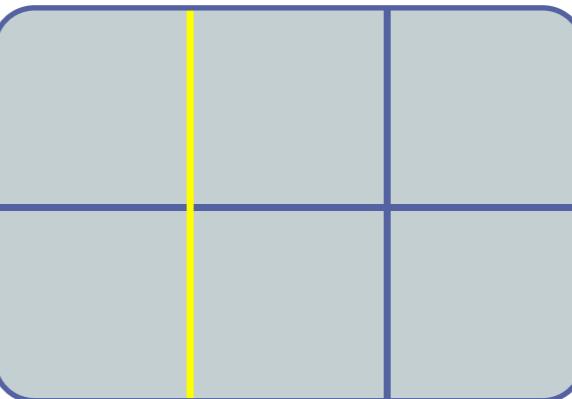
CSS responsive

Responsive met grid

Grid termen

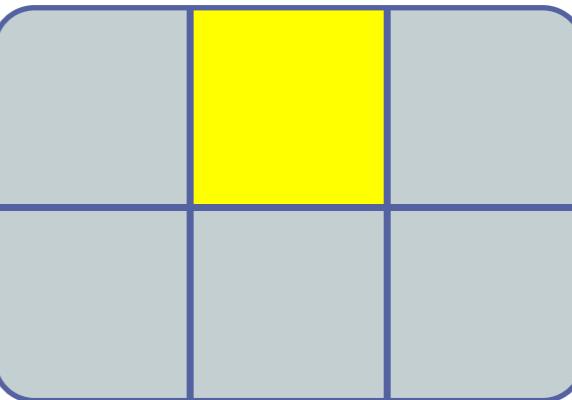
Grid Line

De verticale en horizontale lijnen die de kolommen en rijen van elkaar scheiden.



Grid Cell

De children van de Grid Container. Alleen direct childs van de Grid Container worden Grid Items.



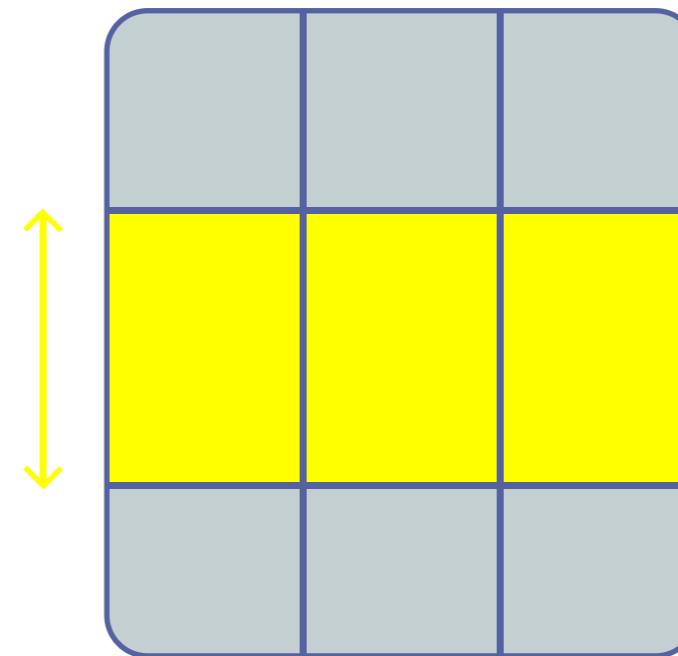
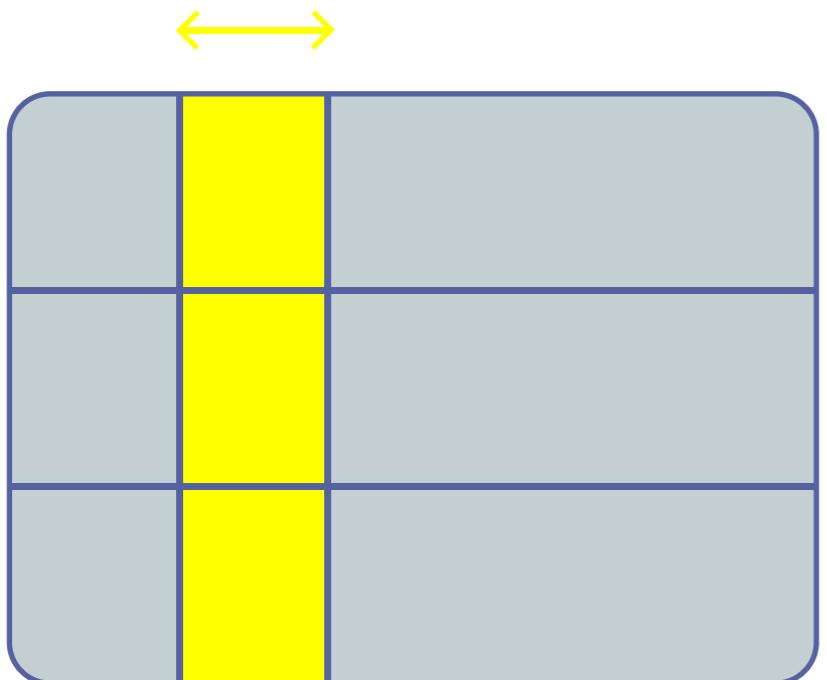
CSS responsive

Responsive met grid

Grid termen

Grid Track

De ruimte tussen aangrenzende Grid Lines.
De hoogte van de rijen of de breedte van de kolommen.



CSS responsive

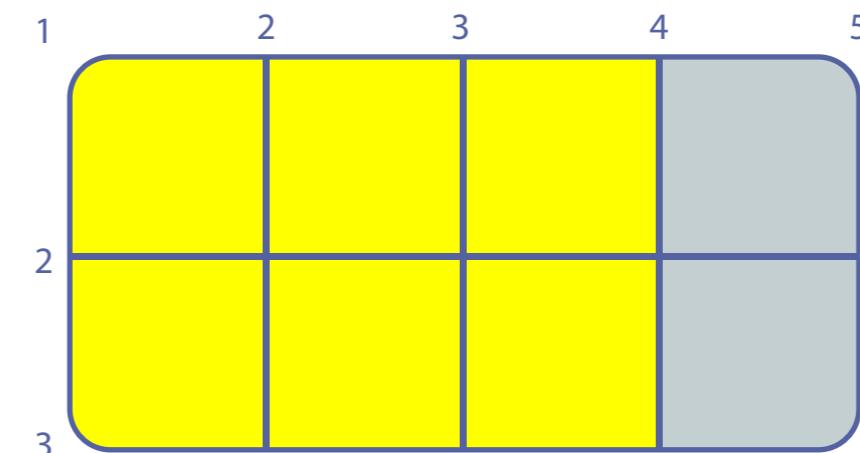
Responsive met grid

Grid termen

Grid Area

Een Grid Area kan over 1 of meerdere grid cellen verdeeld worden. Het is de totale ruimte waar jij een element wil plaatsen omringd door 4 grid lines.

In dit voorbeeld is de Grid Area geplaatst tussen row Grid Lines 1 en 3, en tussen column Grid Lines 1 en 4.



CSS responsive

Responsive met grid

Grid Container Properties

grid-template-columns & grid-template-rows

Definieert de kolommen en rijen van de grid met values gescheiden door een spatie.

De waarden kunnen zijn:

- <track-size> Een lengte waarde, percentage, fractie van de vrije ruimte aangegeven met 'fr'.
- <line-name> Een zelfgekozen benaming voor de lijn.

```
.grid {  
    grid-template-columns: 50% 2fr 1fr 100px;  
    grid-template-rows: [rij1] 25% [rij2] 200px;  
}
```

CSS responsive

Responsive met grid

Grid Container Properties

grid-template-columns & grid-template-rows

Met `repeat` kun je kolommen of rijen herhalen. In de `repeat` geef je aan hoevaak je deze kolom of rij wil maken.

```
.grid {  
    grid-template-columns: repeat(3, 20px);  
}
```

CSS responsive

Responsive met grid

Grid Container Properties

grid-auto-columns & grid-auto-rows

Bepaalt de grootte van alle automatisch gegenereerde rijen of kolommen. Deze worden ge-creëerd wanneer er meer Grid Items zijn dan passen in de Grid.

```
.grid {  
    grid-auto-columns: 200px;  
    grid-auto-rows: 200px;  
}
```

CSS responsive

Responsive met grid

Grid Container Properties

gap

Met gap bepaal je de grootte van de Grid Lines, dit creëert ruimte tussen de Grid Cells. De eerste waarde is de ruimte voor de row-gap de 2e waarde is de ruimte voor de column-gap.

```
.grid {  
    gap: 20px 32px;  
}
```

CSS responsive

Responsive met grid

Grid Item Properties

grid-column-start, grid-column-end, grid-row-start, grid-row-end

Geeft de locatie van een item binnen het grid aan. Aan de hand van de Grid Lines geef je aan waar een item begint en eindigt.

De waarden kunnen zijn:

- <line> kan een nummer of naam zijn die refereert naar een Grid Line.
- span <number> Het Grid Item wijde of hoogte hebben van het aangeven Grid Tracks.

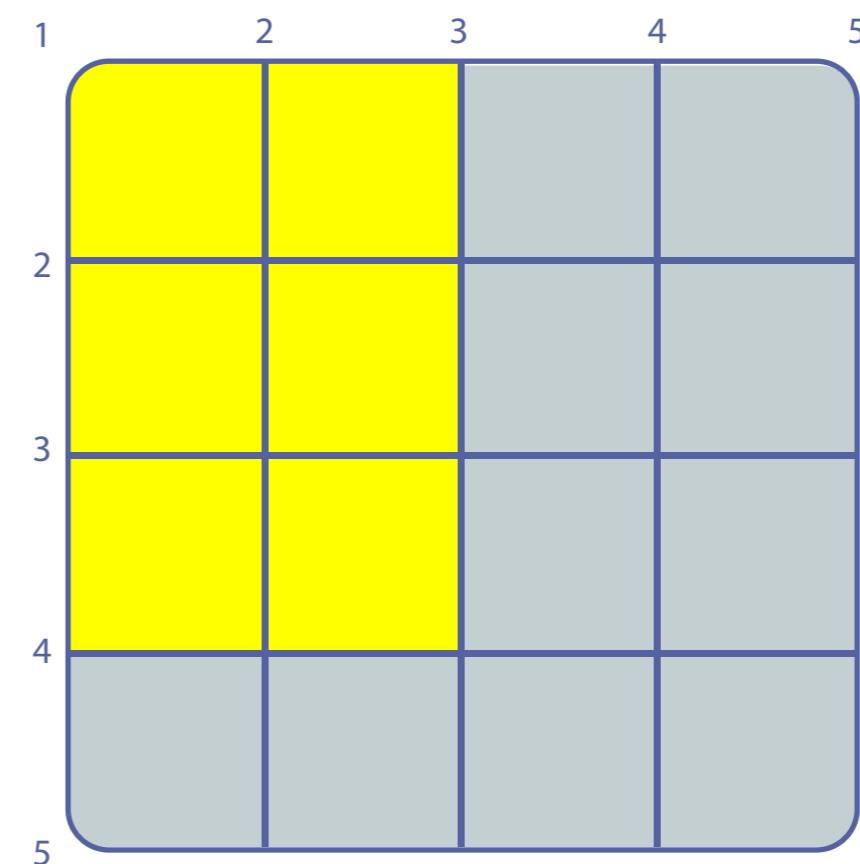
Meer mogelijkheden zie css-tricks.

CSS responsive

Responsive met grid

Grid Item Properties

```
.grid-item1 {  
    grid-column-start: 1;  
    grid-column-end: 3;  
    grid-row-start: 1;  
    grid-row-end: span 3;  
}
```



CSS responsive

Responsive met grid

Aanpassen Grid met min en max width scherm

We kunnen de layout van ons Grid aanpassen door meer of minder kolommen en rijen aan te maken of de hoogte en breedte aan te passen.

Ook is het mogelijk om de plaatsing van de Grid Items aan te passen.

CSS responsive

Opdracht

Maak een contact formulier voor artsy

Het contactformulier bestaat uit een linker en rechter gedeelte. Maak dit met flexbox. Het rechtergedeelte onder de heading ‘send us a message’ is een `<form>` Maak dit gedeelte met Grid.

Maak de pagina responsive. Op kleine schermen zal het blauwe gedeelte onder het formulier komen. En zullen de input velden onder elkaar komen ipv naast elkaar zoals first name last name age email en phone.

Gebruik hetzelfde CSS bestand aangezien veel elementen hergebruikt kunnen worden.

The screenshot shows a website layout for 'Artsy'. At the top, there's a navigation bar with links: 'About us', 'Art gallery', 'Sell yours', and a purple 'Contact' button. Below the navigation, the main title 'Contact Us' is centered. To the left, there's a blue sidebar containing a section titled 'Let's get in touch' with placeholder text and a list of contact details: 'Amsterdam, Grotegracht 12', 'info@artsy.nl', and '010-1234567'. The main content area is titled 'Send us a message' and contains a form with four input fields: 'First Name' (placeholder 'first name'), 'Last name' (placeholder 'last name'), 'Age' (placeholder 'age' with a dropdown arrow), 'Email' (placeholder 'Email'), 'Phone' (placeholder 'Phone'), and a large 'Message' text area (placeholder 'Your message ...'). A purple 'Submit' button is at the bottom of the form. The bottom of the page has a dark footer with the 'Artsy' logo, a tagline 'Happy paintings nascetur ridiculus mus. Donec quam felis.', and links for 'About us', 'Art gallery', 'Sell your art', and 'Contact'.